Schema Mapping Engine: Trade-offs, Optimizations, and Service Architecture Plan

1. Overview

This document outlines the trade-offs made during the development of the Schema Mapping Engine. It provides strategies to address these trade-offs with sufficient computation or development time and defines a modular API-based architecture for scalability and maintainability.

2. Trade-offs and How They Might Be Addressed

2.1 OpenAI Integration via Azure

- **Trade-off:** Used Azure OpenAI instead of direct OpenAI API.
- **Impact:** Increased response latency due to Azure routing overhead.
- **Solution:** Integrate directly with OpenAI API. Enable asynchronous processing and request batching to improve latency.

2.2 Limited Use of token length

- **Trade-off:** Due to token window limitations, the system relies entirely on zero-shot prompting instead of few-shot generalization.
- **Impact:** This reduces the model's ability to generalize across complex or specialized schema mapping tasks, potentially leading to less accurate or less flexible mappings in edge cases.
- **Solution:** Explore alternative approaches to incorporate domain-specific knowledge, such as external context injection or prompt chaining, without exceeding token limits. Additionally, apply advanced prompt engineering techniques to improve the effectiveness of zero-shot prompts.